Capt. George Zolla

Ph.D. Qualifying Examination

April 1996

As in most medium to large institutions, the information system configuration at the Naval Postgraduate School was not specifically architected as it is currently configured, but rather has evolved over the years to meet growing and specialized functional needs. As a result there are islands of automation throughout the School in the form of hundreds of microcomputers and tens of LAN's in addition to the central mainframe computer. The desktop devices are a mix of PC's, Macintoshes and UNIX workstations. These types of information processing platforms are found in business units, administrative offices, and most of the schools and departments throughout the School. In addition, there are stand-alone PCs and Macintoshes used for personal productivity applications. These LANs and stand-alone units are considered by the owning units to be an integral part of information services provided to end-users. Many of them support business applications that duplicate or complement some of the functionality of the central systems.

Some of the data used by these local applications are duplicates of the data maintained on the mainframe with some local enhancements. This data is either entered from the same forms that are sent to the business units for entry, entered from reports generated by the central system, or downloaded from the mainframe system for use by local applications. This duplication is quite costly in terms of personnel, hardware and software. But a more critical issue is the timeliness and accuracy of the information on these local systems as compared to the central site systems, and the difficulty of integrating data from multiple systems and platforms. Since there are inconsistencies between multiple sources of data, a major effort involves reconciliation between the data on the local systems and the data on the mainframe.

1. Propose an architecture for the NPS information system that will provide a framework for making decisions about information systems and for improving the school information system in the future. Make sure your proposed architecture identifies the scope, direction, components (with particular emphasis on the networking and telecommunications infrastructure), and relationships between different components. What is the philosophy and related principles that provide the framework for the proposed architecture?

2. Given the symbiotic relationship between information technology and business process reengineering - information technology enables and empowers users to reengineer business processes, and the reengineered processes determine the need for the information technology - suggest a methodology for business process reengineering using the proposed information system.

3. Propose a plan for migrating from the current systems to the envisioned information system.

INTRODUCTION

In this paper, I will attempt to answer these questions from a conceptual framework. First, I will define the specific information architectural inadequacies at the Naval Postgraduate School (NPS.) I will examine, in detail, two representative examples of the architecture currently in place. Next, I will build a conceptual model that highlights the underlying problems in the present architecture. Third, using existing models and references from information technology readings, I will develop a framework to resolve these conceptual problems. After the framework has been designed, I will develop a specific model for the Naval Postgraduate School. I will then suggest a methodology for business process reengineering using the proposed system. Finally, I will propose a plan for migrating from the current system to the envisioned information system.

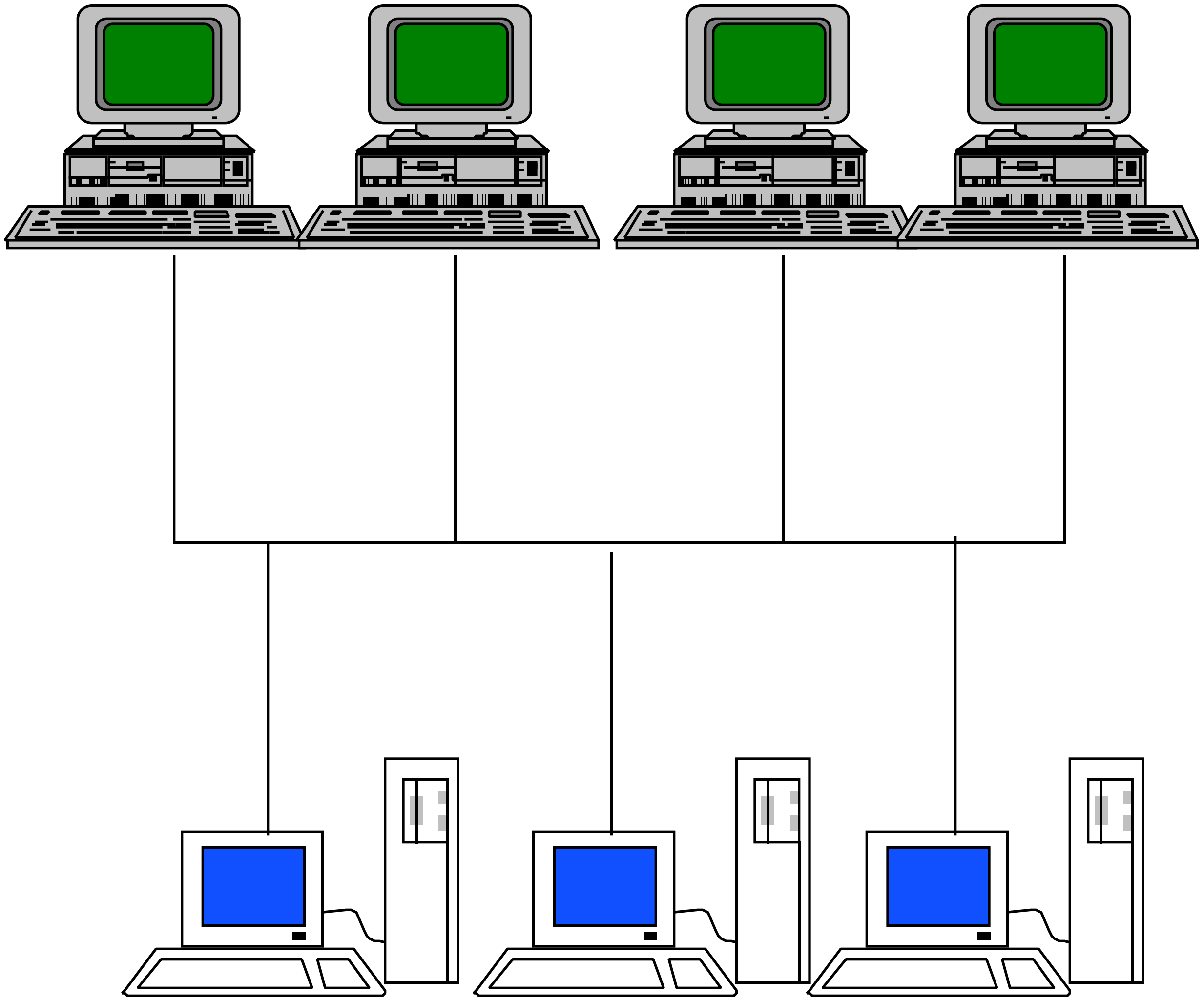
PROBLEM IDENTIFICATION

The NPS information system was originally built around a centralized mainframe Database Management System (DBMS). The DBMS was normally accessed using dumb terminals. When PC LANs grew in popularity, departments created their own file servers that were housed exclusively on their own LAN servers. These LAN servers were normally one of three types based on their network operating system; Intel PC, Mac or UNIX. These LAN servers did not communicate with each other or with the mainframe so multiple sets of data began to appear. As the LAN file servers became more powerful and simpler to manage, departments began creating data that was stored exclusively on their LAN. This lead to the current situation where there are multiple network operating systems supporting multiple LAN file servers that don’t communicate with each other or with the mainframe. There are stand-alone PCs with databases as well as databases stored on Mac, Banyan, Novell, UNIX and soon, NT file servers. This has lead to the present situation where there are numerous databases maintained in different departments that are not easily accessible by other departments or the mainframe DBMS.

I will examine two of the present information systems (IS); the Banyan Vines File System and the mainframe DBMS (FOCUS). Figure 1 depicts the current structure of the Banyan Vines IS; there are multiple hardware servers that house multiple virtual volumes for connected PCs. The Banyan Vines network software connects PCs with these file servers. The PCs do not have access to other LAN’s files unless additional communication software (e.g. TCP/IP) is also loaded. There is no centralized DBMS to coordinate the many database files that are stored on the numerous drives. Normally each individual PC has its own DBMS loaded on its own hard drive (e.g. Paradox or Access.) An additional problem is that some drive letters point to different files in different departments. For instance, drive “N” in one department is not the same as drive “N” in another department. This volume name is used to house specific files for each department. The absence of a network DBMS and the non-connectivity of some drive letters leads to data duplication and inconsistencies of data in the Banyan network. It also creates databases with semantics differences and heterogeneity. Furthermore, PCs with only Banyan Vines proprietary network software are not connected to UNIX, Mac, NT, or mainframes data files.

Figure 2 depicts the mainframe DBMS. The mainframe provides a true DBMS; FOCUS. This encourages database integrity, and consistency. The FOCUS DBMS uses a hierarchical database structure. Use of passwords to access the DBMS provides database security. Connectivity is provided by dumb terminals or 3270 emulation on any campus PCs connected via TCP/IP. File transfer capability to PCs is provided through the mainframe FTP server. Most of the legacy data of the campus is housed in the FOCUS DBMS. Although the mainframe does provide for a campus-wide DBMS there are numerous problems:

Banyan Vines Architecture



File Servers

Figure 1

Mainframe Architecture

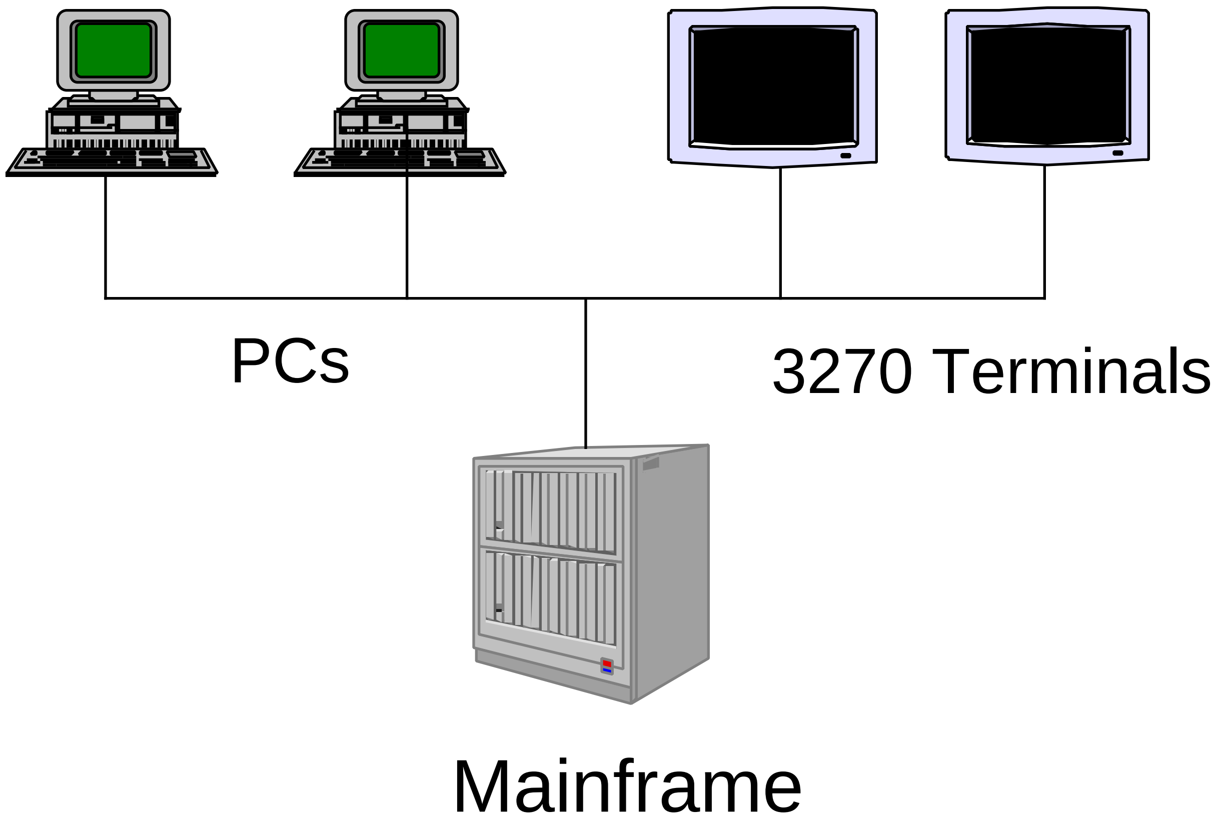


Figure 2

- The hierarchical structure makes it difficult to import/export to relational or flat file systems (There are procedures to save a database file as a flat file).
- Use of dumb terminals or PCs emulating dumb terminals reduces the ability of the end-user to manipulate data and files. The key-mapping on PCs can also lead to confusion.
- The FOCUS DBMS is manipulated by command line instructions that are not advertised as user-friendly. In addition, few users on campus know FOCUS commands.
- To manipulate a database file on your PC, you must first save the file on the mainframe, then run an FTP program to transfer the file from the mainframe to your PC.
- End users must have TCP/IP and 3270 emulation software installed on their PCs to access the mainframe DBMS.
- Because of the hierarchical structure of FOCUS, it is at times difficult to create an equivalent semantic structure on a relational or flat file database.
- Mainframe data files cannot be accessed directly by other campus DBMS.
- If a data file is saved then transferred as flat file to a relational database system or any file server, the data can become inaccurate quickly.
- Because many users on other network systems are unfamiliar with the mainframe environment, they build their own databases and maintain them even when the data is the same as data on the mainframe. This leads to costly duplication of effort and inconsistencies in the data.

CONCEPTUAL MODEL

If we look at several of the IS structures at the NPS, an overall conceptual problem emerges. Figure 3 attempts to depict these conceptual problems that are inherit in the present IS. These problems are not unique to the Naval Postgraduate School. There are many different types of network operating systems running on many different platforms. We have stand-alone systems, Macintosh networks, Banyan networks, Novell networks, UNIX networks and a mainframe system. In addition, Windows NT networks will soon appear. Other organizations may have even more: LINUX, OS/2, SCO. Many times these different systems do not have a centralized DBMS with a enterprise-wide data dictionary and database schemas. These systems do not provide database connectivity to each other. This has lead to islands of automation, data inconsistency and heterogeneity at the DBMS and the semantic level. This semantic heterogeneity is particularly troublesome. It encourages duplication of effort, perpetuation of legacy systems and data inaccuracy. It also inhibits reconciliation between the data on the local systems and the mainframe.

FRAMEWORK

We need an architecture where an end-user from any part of the organization can access enterprise-wide information that is current and accurate. This architecture should enable the user to easily manipulate the requested data files. We could return to the mainframe centralized DBMS paradigm but that doesn’t let us easily manipulate data files, harness the increasing power of the new workstations or use the Graphical User Interface (GUI.) A paradigm that proclaims to enable the user to harness the power of these new systems is the latest and greatest IS architecture, the client/server model. I will assume that the readers are familiar with the general structure and terminology of the client/server paradigm but I would like to begin with a few definitions. Diagrams that are created will be simplified to make the concepts clearer. All servers will be assumed to be DBMS servers with mirrored backup devices.

Client/server is a relationship between processes normally running on separate machines. The server process is the provider of the services. The client is the consumer of the services (Orafi, Harey & Edwards, 1994.) The client and the server are not all that is required. There needs to be

Conceptual Model

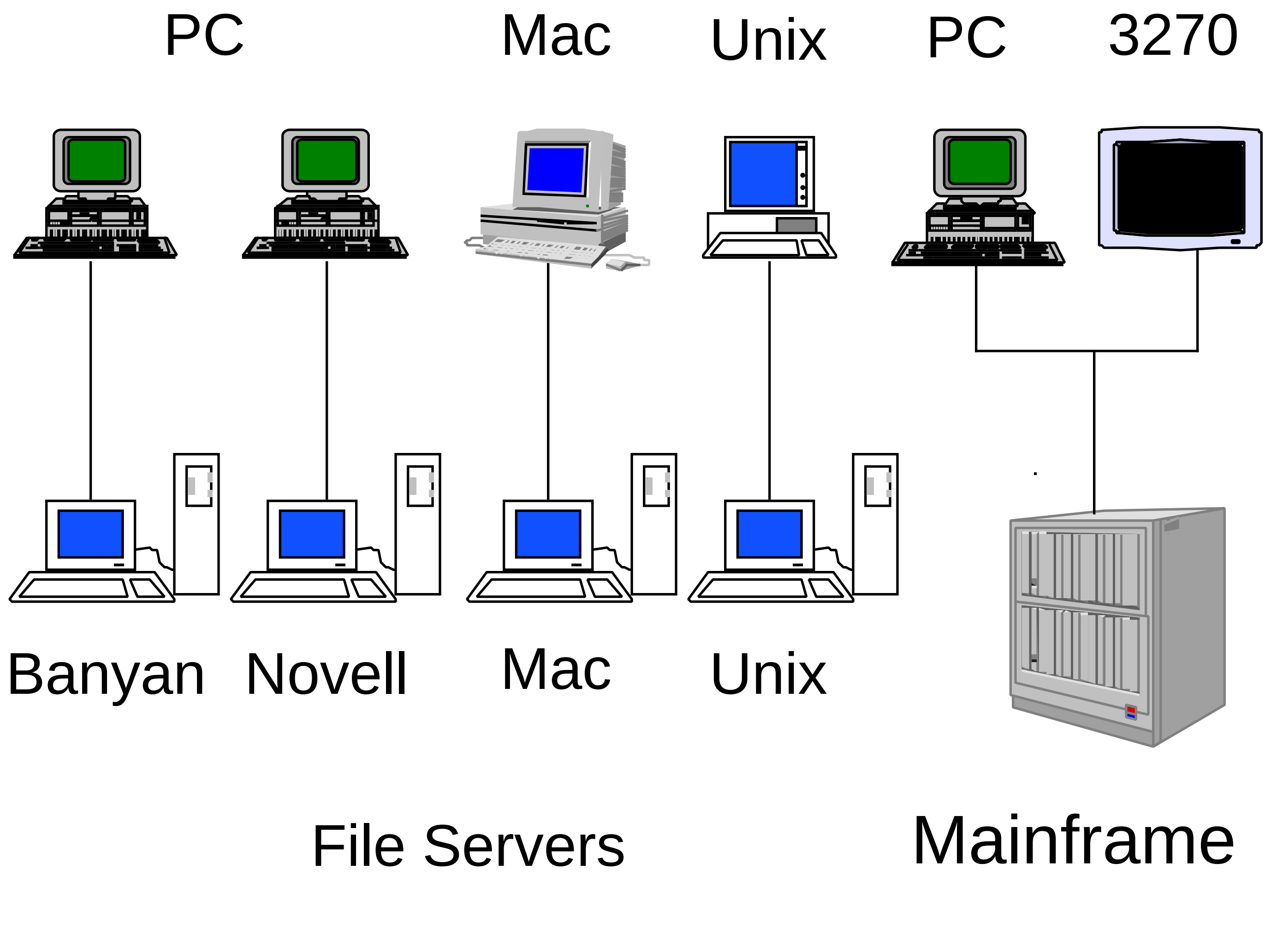


Figure 3

reliable hardware network connectivity, and software to make the connection (middleware.) The architecture theoretically enables any machine in the enterprise IS to connect and manipulate data on the centralized DBMS or to download the data and manipulate it on the client. The architecture is designed to enable any operating system, with the appropriate middleware, to connect to the centralized server. If this architecture works, it will enable all of the machines connected to the enterprise network, with appropriate middleware, access to centralized data and thus correct all the deficiencies in our conceptual model.

Lets take a look at some of the overall characteristics of the client/server architecture. For simplicity we will define the client/server architecture to be in three parts; the client that interacts with the user and server, the middleware which consists of the software required for the connection and the server process that manages the shared resources. The environment is normally heterogeneous and multi-vendor. An extremely important characteristic is horizontal and vertical scalability. Horizontal scalability means adding or removing client machines will produce only a slight performance impact. Vertical scalability means the ability to migrate to larger ands faster server platforms (Kalakota, 1996.)

The client component runs on the users machine's operating system and normally provides a GUI interface. There is client software for almost any operating system (OS) or network operating system (NOS.)

Middleware provides the connectivity so applications can transparently communicate with other programs. This software runs on both the client and server. It normally consists of several layers of software including transport protocol and client-server protocol.

The server component normally runs on a separate machine that has more capacity than the client machines. The faster the CPU, more RAM and the larger storage space the better. The server normally runs a powerful network DBMS or groupware software package. A modern DBMS includes stored procedures and triggers to make the transactions with clients more efficient. Collectively these procedures are sometimes referred to as "TP-lite" (Orafi, Harey & Edwards, 1994.) TP will be defined in the next section.

The smallest client/server environment is the classic two-tier architecture. The client communicates directly with the server with no intervening server. The client can reside on any operating system and any NOS as long as the necessary middleware is available. This works well for small networks with 50 or less clients (Taylor, 1996.) This environment can be difficult to scale up even with larger servers as the number of clients increase and overwhelm the server. Figure 4 depicts a simple two-tier client/server architecture. An example of this architecture is a server running a industrial strength DBMS, the client running any software that can access the DBMS, whether third party or vender specific, and the middleware consisting of a transport stack and vendor specific client-server connectivity software.

Three-Tier Architecture. In a two-tier architecture, the server may become overwhelmed if it receives too many requests from clients. An analogy with ordering a meal is a good one. I have heavily modified this analogy from an article in LAN Magazine (Lazar, 1995.) If you are at home, as a member of a family of six and Dad is cooking breakfast, he can probably remember what you want and remember if the ingredients are available in the kitchen. Here, each member of the family is a client, the father is the server. He uses "TP-lite" for managing the requests for breakfast. Next, imagine arriving at MacDonalds when a bus load of high school kids arrive. If all the customers (clients) went right to the cook there would be pandemonium. Fast-food restaurants have come up with an intermediary between the client and the server, the cashier. The cashier takes the client's orders, determines if the order is on the menu, determines if that order is available, takes the money, gives change and finally gives the client's order to the cook. The cashier obviously takes a major load off the cook. This is what the three-tier architecture attempts to do. Located between the

Two-Tier Client/ Server Architecture

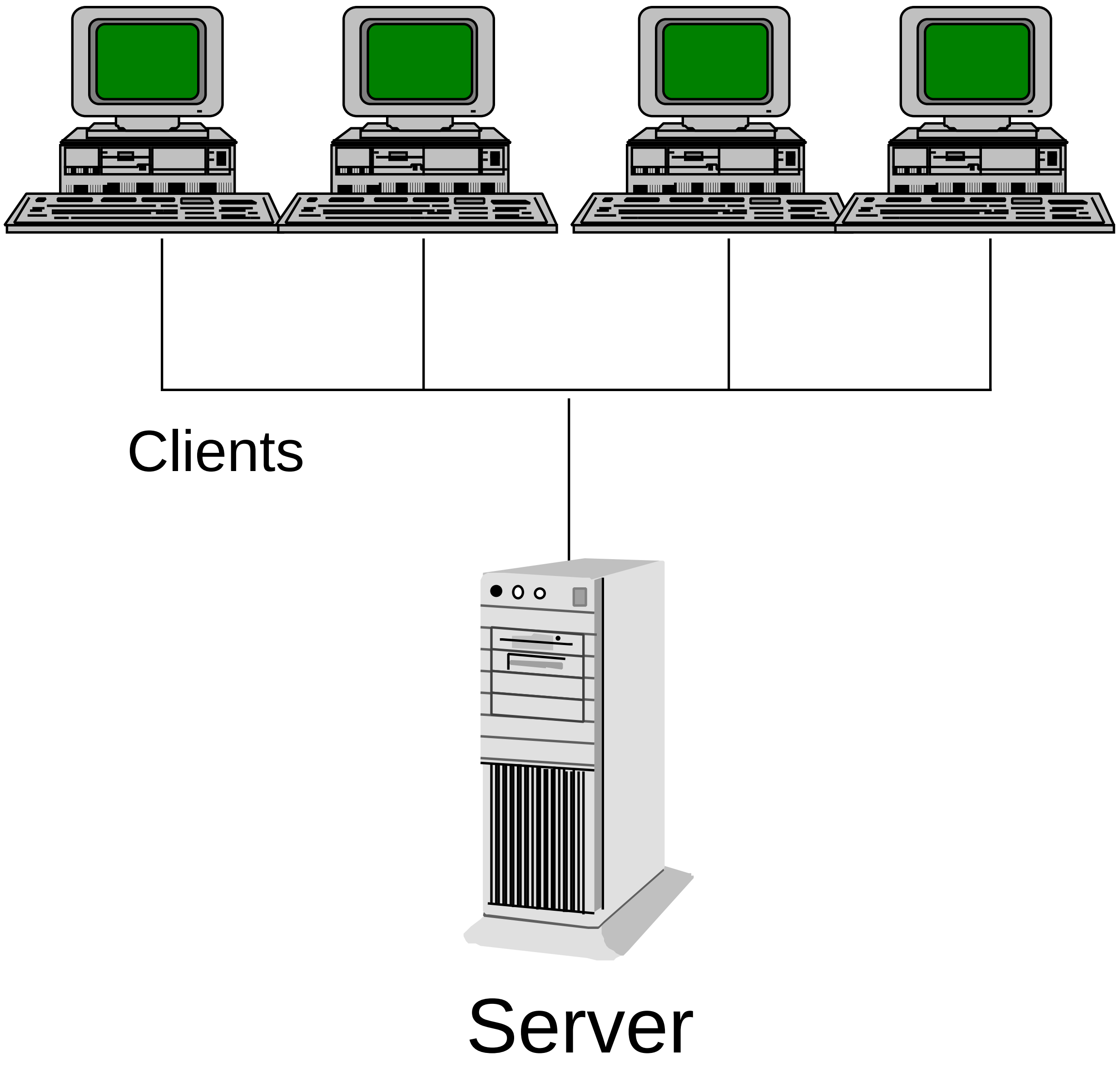


Figure 4

client and the server is a server or agent (figure 5). The agent can provide translation services between legacy mainframe files, limit the number of requests to a given server or map a request to a number of different servers and return a single request to the client. The classic definition of the middle-tier is a Transaction Processing (TP) Monitor. The TP Monitor is typically a separate computer that acts as a operating system for transaction processing. It starts server processes, funnels work to the server, monitors execution, balances the servers workload, ensures the client's access level and more. It is one of the most effective solution for large applications (e.g., 1000 users.) However, there is a downside to TP-monitors, the code to implement them is usually written in a lower level language. It is not available in a 4GL. (Schussel, 1996.) The middle tier can also be the "business process server," that enforces business rules and performs complex processing. From the client's point of view, the business process server is the server. From the database server's point of view, the business process server is the client (Smith, 1995.) I have used the term "TP Monitor" for all middle-tier processes to simplify the diagrams.

The following discussion assumes that the reader is familiar with the basics of web architecture. An interesting service that can be provided by the third-tier agent is that of a web server. Although not a TP Monitor, the classic three-tier component, the web server may also act as an intermediary between the client and server. If the client is using a web browser to access a database, there will be a web server in between the client and server to accept the client request, return an html page if requested, determine access rights and run a Common Gateway Interface (CGI) executable to retrieve information from a database server. The result of the database search, usually a html page, will then be routed to the client. If the web server is connected to a large INTRANET or the INTERNET there could be quite a load on the database server. This situation can be enhanced using a multi-tier client/server architecture (figure 6) where the web server accepts the request from the client and determines if the client needs information from database server. If so, that request is routed to the TP-Monitor for access to the DBMS. This could work quite well, especially if there are a lot of INTRANET or INTERNET applications that use the web browser as the client. CGIs take a lot of CPU cycles. If the load on the web server machine is too high, the CGIs could be placed on a separate machine. This would add another tier to the architecture. This additional CGI server concept is already being implemented in some organizations. It is conceivable that the web browser/web server combination may be the predominate client/server environment of the future. The web browser approach to client/server enables the end-user to work with a familiar interface and normally reduces the complexity of the middleware to just a transport protocol stack with no requirement for a vendor-specific client/server software middleware layer thus reducing costs and complexity. The strength of the web browser/web server concept is that the server and clients use standard open-architecture components to interface with each other.

The best architecture for the individual IS depends on the number of users, types of client software and the amount of load on the server. If the client /server network is small and there are few server requests, the 2-tier approach would probably be the best choice because it is the simplest to maintain and the most economical. If there are many users, heavy server access or there are http-server requests, a three-tier approach may provide the desired responsiveness. If there are many clients and a large amount of http requirements, a multi-tier client/server architecture may be the best choice.

Validation. Validation of these models is beyond the scope of this paper but would be excellent topics for further research.

NAVAL POSTGRADUATE SCHOOL MODEL

I believe that the client/server model is the best architecture for future Information Systems at the Naval Postgraduate School. It provides components (client, server and middleware) that are

Three-Tier Client/ Server Architecture

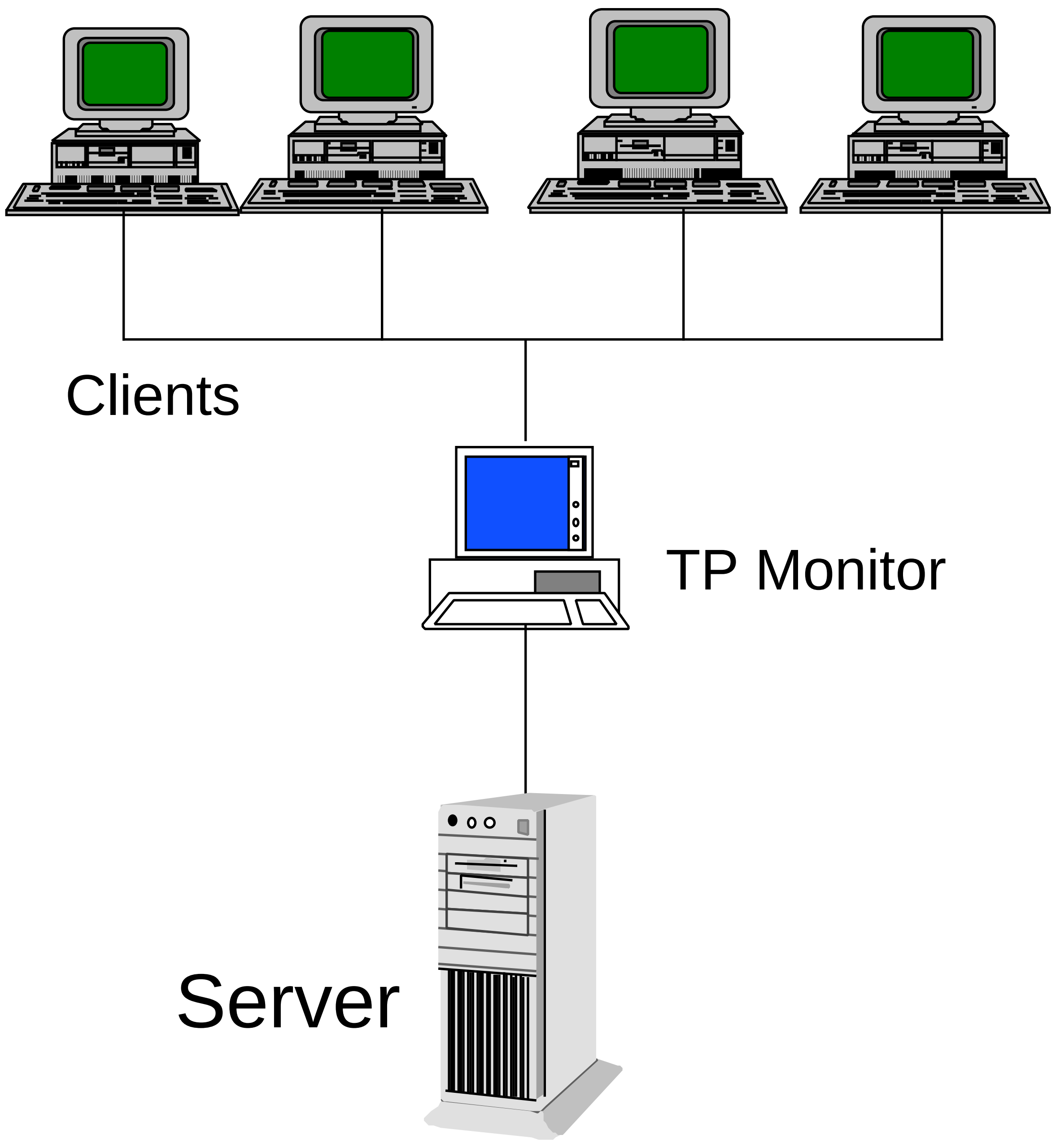


Figure 5

Multi-Tier Client / Server Architecture

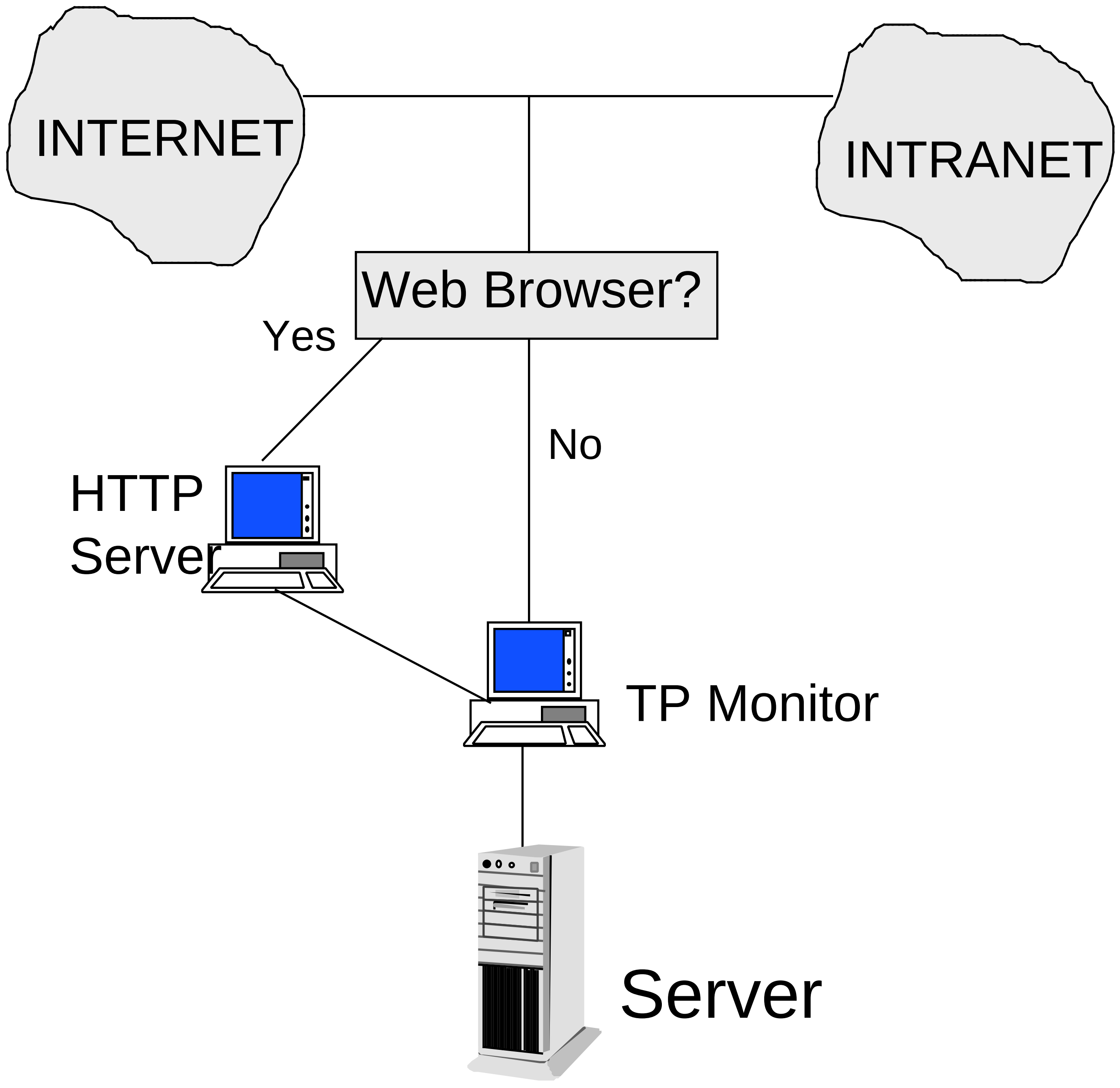


Figure 6

scaleable and can be used to provide many different combinations of information services depending on the present and future requirements of the school. This architecture not only provides for future growth, but it also takes advantages of the increasing power of client workstations, and treats data as a an enterprise-wide asset that should be available to everyone within the organization.

The choice of a two, three or multi-tiered client/server architecture should be determined by the requirements of the school. Because the NPS has over 2000 computers connected to its internal network (INTRANET) the two-tier approach would probably be slow and ineffective. The three-tier approach appears the to be the best option because of the number of potential clients, hardware scalability and the ability to move to a multi-tier approach if the web browser becomes an important client software. The central client/server database server should be a relational DBMS because it would be able to host almost all the non-mainframe legacy database files. In addition, the middle-tier machine could be a combination TP Monitor and M/F hierarchical DBMS translator that would make legacy hierarchical database files available (Taylor, 1996.) There are also relational/hierarchical DBMS translators that could be used to provide a direct link between the relation DBMS and the mainframe DBMS (Oracle White Paper, 1995.) The choice of translating the FOCUS DBMS files on the mainframe or with a middle-tier server needs further research to find the best solution. Figure 7 depicts the recommended NPS IS architecture linking all present end-users except dumb terminals to the centralized DBMS. A possible future NPS C/S architecture that incorporates support for web browsers from the INTRANET or INTERNET and anticipates the demise of the mainframe DBMS would be identical to figure 6. If the mainframe DBMS is still an important part of the IS structure, it can be incorporated into the future architecture as a component as in figure 7. Either of these possible future architectures are scaleable from the recommended NPS architecture.

The real asset of the client/server architecture is the ability to combine the basic components in numerous combinations to optimize the local system. For instance, additional database servers can be added, additional TP Monitor or Business Rules servers can be added to enhance DBMS performance. The client/server architecture gives the NPS the flexibility to meet future user requirements in an efficient, timely manner.

BUSINESS PROCESS REENGINEERING METHODOLOGY

Business process reengineering (BPR), is a theoretical way of redesigning a process important to customers from the ground up. Not all or even most BPR moves are successful. It is vital to be clear on what process to reengineer and make sure that everyone understands the business case to change. Success in BPR requires a mandate from upper management. (Ovans, 1995.)

We need to consider our fundamental business processes and decide which ones will produce significant improvements for our customers using the new information structure. To do this we will need to train our personnel on the basics of reengineering principles, have them look at the new IS and develop plans to change the processes we use. Robert Janson (1992, National Productivity Review) promotes the concept of the three “R”s of reengineering; Rethink, Redesign, Retool. If we assume that the proposed NPS IS retools our information structure, then the we need to concentrate on rethinking and redesigning our business processes.

Rethink. The new paradigm that we need to use is the concept of centralized information. The members of the NPS need to think of data as critical to the success of organization. The concept of entering data only once to increase overall efficiency and accuracy needs to be a fundamental component of how we do business. With the new IS organization structure, shared database information will be a reality that can be incorporated into numerous business processes.

NPS Model

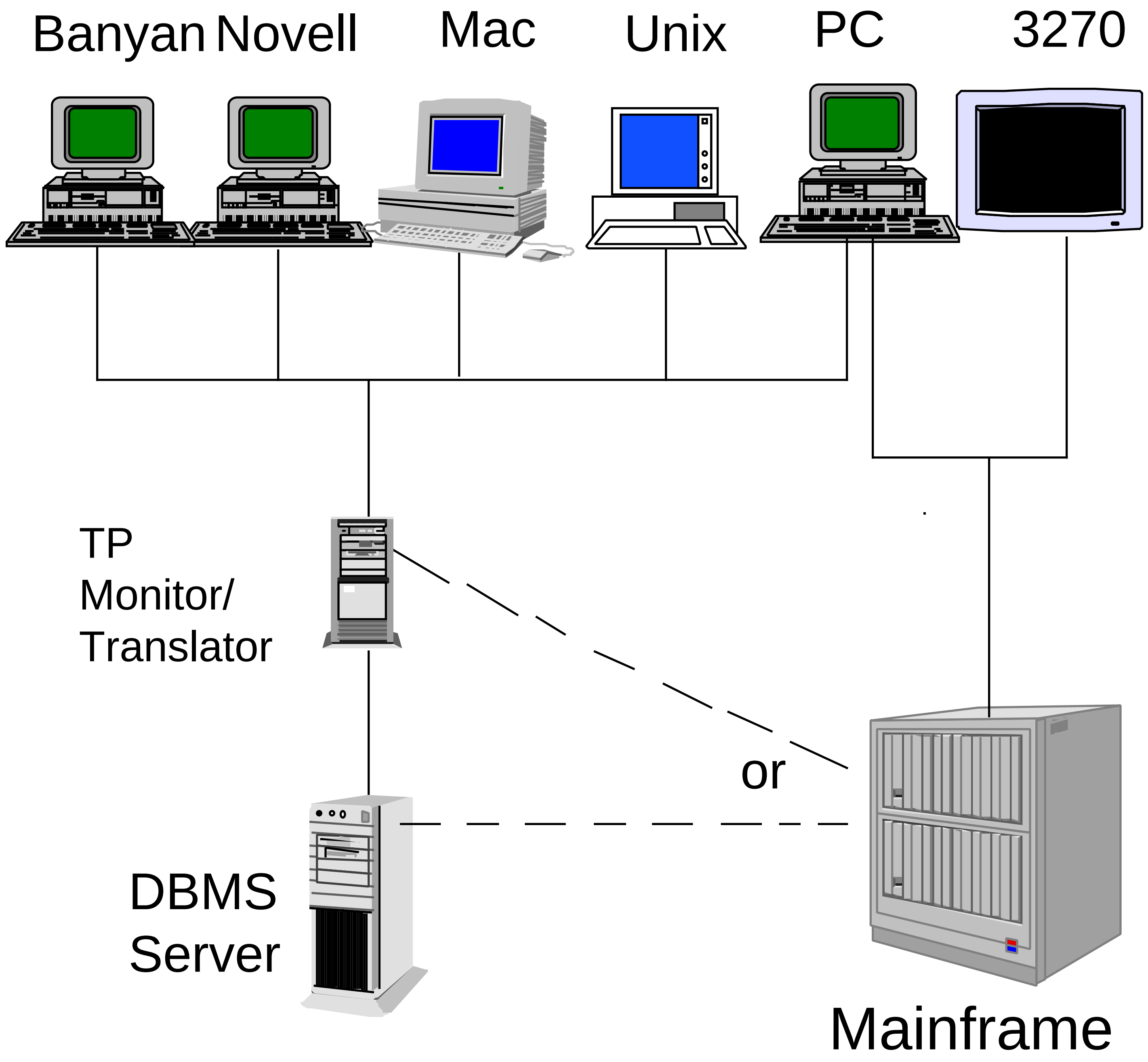


Figure 7

With these basics introduced, we need to select business processes that can most productively use our new IS. Upper management must take the lead in advocating that departments work together to review business processes to look at ways we can be more productive. We must analyze what products and services we produce. We need to identify the steps we use in accomplishing these tasks. An example might be student information. How is the information entered for new students? How many times and what different databases is that information entered? Why is it done that way? Which departments are involved? How many times is the same data entered into different non-sharable databases?

Redesign. BPR teams are created for selected processes. Users and providers of the process should be involved as well as members of the departments responsible for successful completion of the process. A few trained Information Technology (IT) professionals should also be included. The IT professionals could come from inside or outside the organization. There are good points for both approaches. Outside personnel would be expected to have new ideas, experience at BPR and the have the latest technical expertise. Personnel from inside the NPS organization have an advantage in that they already know the people and the structure of the organization. Theoretically, with BPR, you can create an entirely new way to conduct some process that's important to your company's customers. You must first be clear on what process is going to be reengineered. A simplified version of structured system design can be used to create objects, look at data flow and define required processes. CASE (Computer Aided Software Engineering) can be a great help in graphically conceptualizing the process. CASE tools can even be used to determine the schema of database files and create code for parts of the reengineered process.

A five-step approach to BPR (figure 8), using the techniques we have developed puts the overall process into a logical sequence (Davenport & Short, 1990.) Each of these steps must be supported by top level management for the process to succeed.

We must concentrate on effectively using the new IS as an integral part of the reengineered process. Specifically determine how a shared database can reduce data entry, reduce user involvement and improve the overall process. Include in the process discussions and demonstrations on how the use of a shared database can help increase efficiency in other processes that are done in that department, possibly for future consideration in BPR.

Select one process that uses the new IS and appears to be a real winner. Use the Rapid Prototyping Method to create an application as quickly as possible to show the value of BPR. If the value becomes clear, this will spur efforts on many levels to embrace the BPR concept throughout the organization.

MIGRATION PLAN

To develop a migration plan let’s start with the basic components of the client/server architecture. We originally defined the architecture into four parts: the client, network, connection software (middleware) and the server. A good migration plan from the present non-integrated architecture to a true client/server architecture will require that all these components be in place. It is critical for the successful transition to the client/server architecture that an enterprise wide client-server plan for everything from platforms and applications to tools, security procedures, backup, training (of both users and IS staff), and infrastructure support be in place. It's critical to get both executives and end users to understand and buy into the plan; the most successful way to do that is to involve them in developing the plan in the first place (Diamond, 1995).

Business Process Redesign

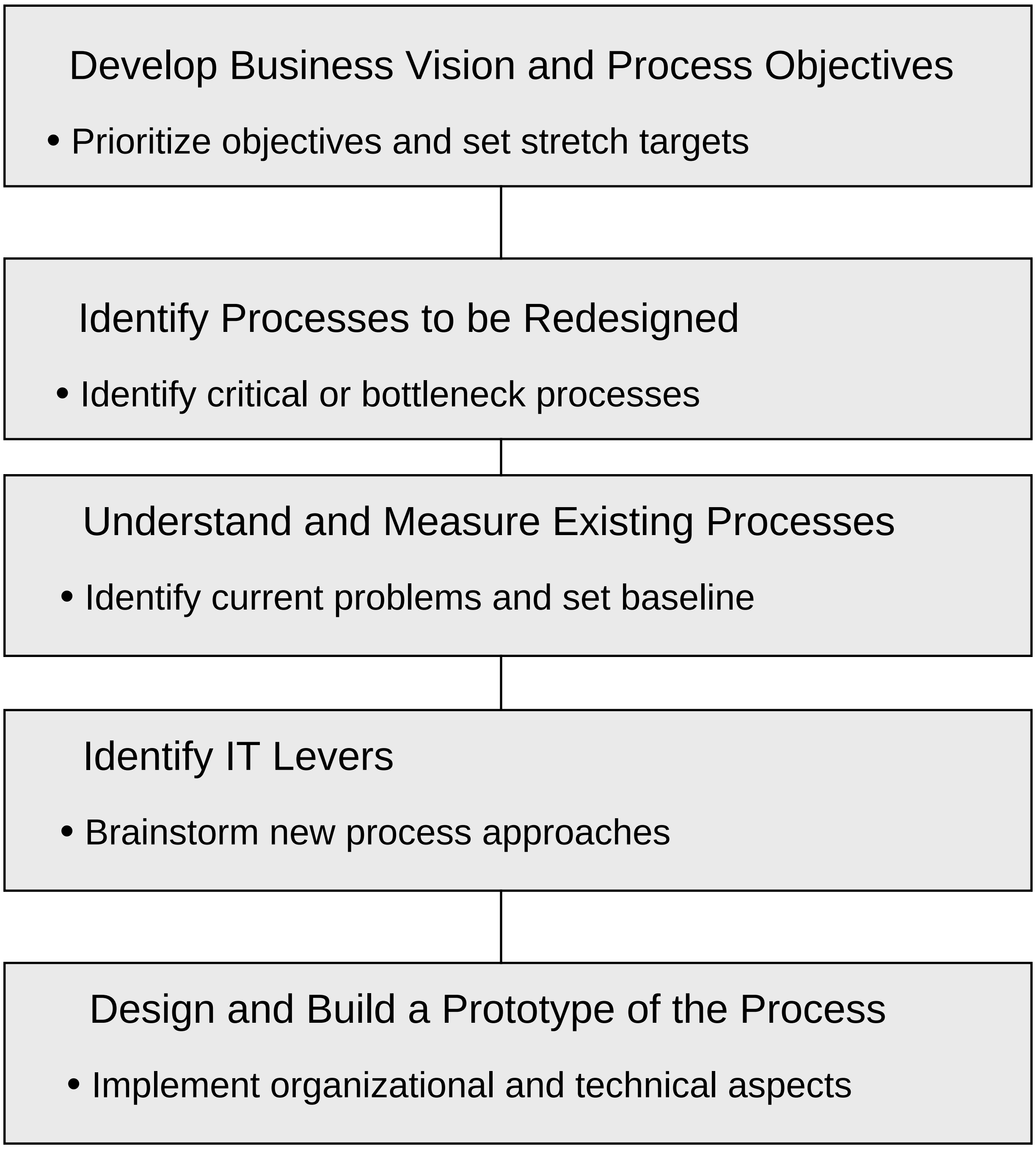

Develop Business Vision and Process Objectives
• Prioritize objectives and set stretch targets
Identify Processes to be Redesigned
• Identify critical or bottleneck processes
Understand and Measure Existing Processes
• Identify current problems and set baseline
Identify IT Levers
• Brainstorm new process approaches
Design and Build a Prototype of the Process
• Implement organizational and technical aspects

Figure 8

On the ground floor of this structure is network connectivity. Good, fast, reliable, well-maintained network connectivity needs to be in place before anything else will work. This requires highly reliable network wiring, high speed hubs, routers and gateways. Highly trained personnel, adequate in number for seven day/twenty four hours a day coverage is required. Hardware testing equipment is needed to maintain the network reliability and troubleshooting. A plan should also be in place for future speed enhancements.

Next, client and server platforms and their software need to be examined together. We already have multiple client operating systems in place on multiple hardware platforms. Therefore client software that is multi-platform capable is required.. The server hardware platform/operating system and database management system is the heart of our new architecture. A very powerful multi-tasking operating system and hardware platform is required. A hardware/operating system combination that is familiar to the current network maintenance staff would be beneficial. Middle-tier server software/hardware needs to be selected that can access the mainframe database, prioritize requests from clients and provide connectivity to the central DBMS. The central DBMS must be powerful, able to communicate with multi-operating systems and have sufficient middleware products available to make it very flexible to our changing needs. I believe that at present a relational database model would give the most flexibility.

Middleware software to provide connectivity from the clients to the server is the final component. There are normally several layers of software that make up the total middleware component. The transport protocol, Network Operating System (NOS) and service specific software make up some of these layers. Connectivity for all clients and all services to all servers must be made provided as well as avoiding middleware layer incompatibilities.

Planning and implementing a client/server system can be more difficult than it appears (Lumbley, 1995.) The real secret to client/server success today is to keep it simple. “Pick a single client/server platform, limit the number of vendors, limit mix-and matching, keep the project simple, use a single server, and get the system up and running fast.” (Orafi, Harey & Edwards, 1994.) With this said, it is hard to find fully integrated turn-key systems. Most products are single component pieces. At present, it is very difficult to integrate multi-vendor, multi-platform, multi-operating system applications and produce a fully integrated low maintenance client/server information system.

Training people to use and maintain these new systems will be a time consuming, expensive undertaking. In some organizations the cost of hardware and software for client-server computing is trivial compared with the cost of training. Users need training not only in using the new applications, but IT professionals need training to maintain the more sophisticated DBMS, additional servers and the additional middleware required.

While the architecture is being developed an in-depth analysis should be undertaken to develop a campus-wide database schema that can be used for all shared data. Legacy data is an important asset to the NPS. Access to that data whether it is on the mainframe DBMS or on a Banyan database file is crucial to the success of the transition to any new system. Heterogeneity exists at three levels: platform, DBMS and database semantics (Kamel, 1995). A major effort must be conducted to make the present data available to all clients. This present heterogeneity highlights an exceptional opportunity that the envisioned client/server architecture can give us - the establishment of a campus-wide data dictionary to ensure compatibility of future data structures and definitions. This will significantly contribute to the sharing of enterprise-wide data.

The overall structure for the implantation plan should:

a. Involve end-users, executives and information systems specialist in developing the overall implementation plan. Education for the participants on client/server theory is mandatory.

b. Use the Naval Postgraduate School Model, as detailed above, as the framework for the future architecture.

c. Ensure that each component of the client/server model is in place.

d. Define a campus-wide data dictionary to include semantic definitions and database structures.

e. Include a massive training plan for all of the users and maintenance personnel to ensure that end-users know how to use the new architecture and that the new information system is reliable.

SUMMARY AND CONCLUSIONS

The client/server architecture is a clear winner as a method to provide an integrated information system for the Naval Postgraduate School. It will enable clients to access enterprise-wide information, reduce redundant and inconsistent data and give tremendous capabilities and flexibility to end-users. However, all these advantages do not come without a price. The client/server architecture is at present complex and expensive. An overall enterprise-wide plan for the transition to the client/server architecture must be researched in-depth before proceeding. As the new structure is put in place, training must be conducted for the IS professional staff and end-users to fully utilize the potential of this new architecture.

Client/server architecture appears to be the wave of the future. It increases our capabilities today and sets up an open-architecture framework for future clients, middleware and servers. It also gives a firm foundation for the object-oriented world that is on the horizon. Lets get started!

References

Davenport, Thomas H. & Short James E.(1990). *The New Industrial Engineering: Information technology and Business Process Redesign*. Sloan Management Review. Summer

Diamond, Sid (1995). *Client-server: the myths.* Information Week. Dec 18, n558 p136(1).

Janson, R. (1992) *How Reengineering Transforms Organizations to Satisfy Customers.* National Productivity Review.

Kalakota, Ravi. (1996). *What are the characteristics of client/server architecture?.* http://www.cis.ohio-state.edu/hyper...t/client-server-faq/faq-doc-13.html.

Kamel, Magdi (1995). *Identifying, Classifying, and Resolving Semantic Conflicts in Distributed Heterogeneous Databases: A Case Study*. Journal of Database Management. Winter, Vol. 6 No. 1, pg 21.

Lazar, Bill. (1995). *Client-server gets serious*. LAN Magazine Oct p123(4).

Lumbley, Joe (1995). *Planning client-server systems.* Data Based Advisor Oct v13 n9 p40(5).

Oracle White Paper (1995) *Mainframes and Open Systems Cooperating in a Distributed Enterprise.* Oracle Corporation.

Oracle White Paper (1994) *Oracle Transparent Gateway for EDA/SQL Product Review.* Oracle Corporation.

Orfali, R.; D. Harkley; & J. Edwards. (1994). *Essential Client/Server Survivor Guide.* Van Nostrand Reinhold.

Ovans, Andrea (1995). *Should you take the reengineering risk?* Datamation Sept 15, v41 n17 p38(4).

Schussel, George (1996). *Client/Server, from dBase to Java: Is it over, or just Beginning?* http://www.dciexpo.com/geos/java.htm

Smith, Brian J. (1995). *Three-tiers for client-server.* Data Based Advisor Oct v13 n9 p40(5).

Taylor, Lloyd. (1996). *What is a Three-Tier Architecture?* http://www.cis.ohio-state.edu/hyper...t/client-server-faq/faq-doc-8.html.